WORKMAN NYDEGGER
A PROFESSIONAL CORPORATION
ATTORNEYS AT LAW
1000 EAGLE GATE TOWER
60 EAST SOUTH TEMPLE
SALT LAKE CITY, UTAH 84111
TELEPHONE (801) 533-9800
FAX (801) 328-1707

FROM: Rick D. Nydegger

TO: Supervisory Examiner Tuan Q. Dam

RE: Proposed Agenda for Interview – SN10/618,919 (formerly assigned to Examiner Andrew Y. Chou)

DATE AND TIME: Friday, July 11th 3:30pm

Proposed 30 Minute Interview

I. Brief review of status of case, and nature of the invention

II. Review of rejections of record and prior art applied in the current Office Action

III. Presentation of proposed claim amendments and explanation of how those amendments are believed to overcome the rejections and prior art of record

IV. Discussion with Examiner and any further amendments resulting from the Examiner's views and comments, with the objective of reaching claim amendments which fully address and resolve the issues of record and advance the case over the prior art of record

V. Preparation of Interview Summary

ELECTRONICALLY FILED PATENT APPLICATION

Docket No. 13768.1073

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of		)
	Aaron Cornelius et al.	)
Serial No.:	10/618,919	) Art Unit
		) 2192
Filed:	July 14, 2003	)
Conf. No.:	7973	)
For:	DYNAMIC CONTEXTUAL HELPER USER INTERFACE	)
Examiner:	Andrew Y. Chou	)
Customer No.:	047973	)

AMENDMENT "D" AND RESPONSE

Mail Stop AMENDMENT
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

In response to the Office action mailed February 27, 2008, please amend the above-identified application as follows:

Amendments to the Specification begin on page 2 of this paper.

Amendments to the Claims are reflected in the listing of claims which begins on page 4 of this paper.

Remarks/Arguments begin on page 11 of this paper.

AMENDMENTS TO THE SPECIFICATION

Please replace paragraph [02] with the following marked-up version of the paragraph:

[02] Computer applications that include visual design surfaces are commonly used to generate artifacts such as computer code or to automate processes. On a visual design surface, these artifacts are typically represented by shapes. The shapes [[are]] placed on the visual design surface are often arranged in a particular order and connected together in a particular fashion to perform specific functions. Each of the shapes also typically includes configuration parameters that are set by the user. After the user lays out the pattern of shapes and sets the appropriate configuration parameters, the design is compiled.

Please replace paragraph [03] with the following marked-up version of the paragraph:

[03] During the compilation process errors are identified and presented to the user. For example, if a variable has not been initialized a compilation error identifying this error is presented to the user. There are several disadvantages in waiting to identify errors until the compilation process. One disadvantage is that with large designs, it can be difficult for a user to remember the intended function or configuration of [[a]] <u>an</u> artifact at a later time.

Please replace paragraph [25] with the following marked-up version of the paragraph:

[25] Each shape in design section 202 includes one or more configuration parameters. As the user interacts with the design surface, configuration parameters are checked to ensure that they are in compliance with a predetermined set of rules. For example, send shape 204 may include a configuration parameter for identifying a protocol to use when sending data to port shape 206. A rule may require the protocol configuration parameter of send shape [[202]] <u>204</u> to match the protocol used by port shape 206. The predetermined set of rules may relate to the context in which a shape is being used. For example, a set of send rules may be used when a send/receive shape is used to send data. When the rule is violated, an inconspicuous icon, such as icon 212 may be displayed next to the shape to alert the user.

Please replace paragraph [29] with the following marked-up version of the paragraph:

[29] When a configuration rule is violated, in step 408 an icon is displayed next to the relevant shape or shapes. Figure 5A illustrates a suitable inconspicuous error icon 502. Error icon 504, shown in Figure 5B, represents a rollover state of icon 502. When a cursor is positioned over icon 502, icon 502 is replaced with icon 504. Alternatively, icons may be activated with keyboard commands. One advantage of having two different states is to alert the user that there is an actionable behavior on the relevant shape. Displaying a drop down arrow alerts the user that a drop down menu will be displayed when the icon is selected. Error icons 502 and 504 may be placed at the edge of a shape to minimize interfering with the design. Similarly, the icons may be placed next to the relevant shape. In step 410 it is determined whether the user has selected the error icon. When the user selects the error icon, in step 412 at least one proposed solution to the configuration error is displayed to the user. Figure 5C shows two proposed solutions 506 that are displayed to the user. A proposed solution may include a dialogue box. The dialogue box may prompt the user to initialize a variable or set a configuration parameter. ~~Of course text boxes, drop-down lists, buttons and other user interface controls may also be~~ Of course text boxes, drop-down lists, buttons and other user interface controls may also be placed in the drop-down menu itself, avoiding the need to launch a dialog or wizard, and allowing the user to perform editing in place. Other proposed solutions include creating a new design element, adding or connecting a shape, etc. One skilled in the art will appreciate that numerous additional and alternative proposed solutions may be used. The proposed solutions may be a function of the context of the design.

AMENDMENTS TO THE CLAIMS

This listing of claims replaces all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) In a computing system which includes a visual design surface in the form of a user interface having a plurality of shapes that are selectable by a user, each shape being associated with one or more configuration parameters that define characteristics of software components such as relative positioning of shapes, connections between shapes and other design parameters set by a user, and wherein a user may select and arrange a plurality of the shapes when designing a software application by dragging and dropping them to a design section of the interface, a method of alerting the user to configuration errors that may arise due to inconsistencies between the configuration parameters of one or more of the selected shapes for the software design, the method comprising:

    the user selecting a first shape and copying it to the design section of the interface;

    the user selecting a second shape and copying it to the design section of the interface so that the second shape is functionally interactive with the first shape;

    automatically evaluating with a configuration module the configuration parameters of the first and second shapes by accessing a configuration rules database which is used by the configuration evaluation module to determine whether the configuration parameters of the two functionally interactive shapes in the proposed design violate any configuration rules; and

    when a configuration rule is determined to have been violated, automatically evaluating any errors using an error module that accesses a common error database which defines at least some errors as being grouped under a common error, and then returning a common error for display to the user in order to limit the number of error messages presented to the user during the design process;

    displaying an icon next to a at least one of the first or second selected shapes to represent at least one common configuration error associated with the shape; and

    in response to a user selecting the icon, displaying at least one proposed solution to the common configuration error presented.

2. (Canceled) The method of claim 1, further including:

(c) comparing shape configuration parameters of the shape to configuration parameter rules.

3. (Currently Amended) The method of claims 1 or 31, wherein the configuration parameter rules are selected based on a context in which the shapes are being used.

4. (Currently Amended) The method of claims 1 or 31, further including:

removing the icon when the user has taken one or more actions and the common configuration error no longer exists.

5. (Canceled) The method of claim 2, wherein (c) is repeated periodically.

6. (Canceled) The method of claim 5, wherein (c) is repeated when at least one configuration parameter of the shape changes.

7. (Canceled) The method of claim 6, wherein (c) is repeated when at least one configuration parameter of a shape other than the shape in (a) changes.

8. (Currently Amended) The method of claims 1 or 31, wherein the at least one proposed solution is presented in a dialog box associated with the icon.

9. (Original) The method of claim 8, wherein the dialog box prompts a user to initialize a variable.

10. (Original) The method of claim 8, wherein the dialog box prompts a user to set a configuration parameter.

11. (Currently Amended) The method of claims 1 or 31, wherein the at least one proposed solution comprises a wizard.

12. (Currently Amended) The method of claims 1 or 31, wherein the at least one proposed solution comprises creating a new design element.

13. (Currently Amended) The method of claims 1 or 31, wherein the at least one proposed solution comprises adding a shape.

14. (Currently Amended) The method of claims 1 or 31, wherein the at least one proposed solution to the common configuration error comprises adding a necessary shape that is not yet included in the design section.

15. (Currently Amended) The method of claims 1 or 31, wherein the at least one common configuration error comprises configuration parameters that are set by the user in an inconsistent manner.

16. (Original) The method of claim 15, wherein the inconsistent configuration parameters are configuration parameters of the same shape.

17. (Currently Amended) The method of claim 15, wherein the inconsistent configuration parameters are configuration parameters of the first and second shapes.

18. (Currently Amended) A method as defined in claims 1 or 31 wherein at least one of the shapes is a container shape.

19. (Currently Amended) The method of claim 18, further including:

(c) expanding the container shape to display at least the shape contained within the container shape; and

(d) displaying the icon next to a shape contained within the container shape and that contains the at least one common configuration error.

20. (Currently Amended) The method of claim 18, wherein the at least one common configuration error comprises a necessary shape that is not connected to the shape contained within the container shape.

21. (Currently Amended) The method of claim 18, wherein the at least one common configuration error comprises configuration parameters set by the user in an inconsistent manner.

22. (Original) The method of claim 21, wherein the inconsistent configuration parameters are configuration parameters of the same shape.

23. (Currently Amended) The method of claim 21, wherein the inconsistent configuration parameters are configuration parameters of the first and second shapes.

24. (Canceled) A method of alerting a user of configuration errors of shapes representing software artifacts and displayed on a visual design surface, the method comprising:

(a) comparing shape configuration parameters of a shape to configuration parameter rules to identify configuration errors;

(b) determining a common error that causes the identified configuration errors;

(c) displaying an icon next to a shape to represent the configuration errors; and

(d) in response to a user selecting the icon, displaying at least one proposed solution to the common error.

25. (Canceled) The method of claim 24, wherein the at least one proposed solution comprises setting a configuration parameter.

26. (Canceled) The method of claim 24, wherein the at least one proposed solution comprises creating a new design element.

27. (Canceled) The method of claim 24, wherein the at least one proposed solution comprises adding a shape.

28. (Canceled) The method of claim 24, wherein the common error comprises a necessary shape that is not connected to the shape in (a).

29. (Canceled) In a computer system having a graphical user interface including a display and a user interface selection device, a method of indicating configuration errors of elements displayed on a visual design surface and representing software artifacts, the method comprising:

(a) displaying a plurality of the elements on the design surface;

(b) displaying an icon next to an element to identify a configuration error associated with the element; and

(c) in response to a command from the interface selection device, displaying at least one proposed solution to the configuration error.

30. (Canceled) A computer readable medium containing computer executable instructions for causing a computer system to perform the steps comprising:

(a) displaying on a design surface a plurality of shapes representing software artifacts;

(b) displaying an icon next to a shape to represent at least one configuration error associated with the shape; and

(c) in response to a user selecting the icon, displaying at least one proposed solution to a configuration error.

31. (New) In a computing system which includes a visual design surface in the form of a user interface having a plurality of shapes that are selectable by a user, each shape being associated with one or more configuration parameters that define characteristics of software components such as relative positioning of shapes, connections between shapes and other design parameters set by a user, and wherein a user may select and arrange a plurality of the shapes when designing a software application by dragging and dropping them to a design section of the interface, a computer program product for implementing a method of alerting the user to configuration errors that may arise due to inconsistencies between the configuration parameters of one or more of the selected shapes for the software design, the computer program product comprising:

a computer readable medium comprised of a computer storage medium containing executable instructions for implementing the method, and

wherein the method is comprised of:

the user selecting a first shape and copying it to the design section of the interface;

the user selecting a second shape and copying it to the design section of the interface so that the second shape is functionally interactive with the first shape;

automatically evaluating with a configuration module the configuration parameters of the first and second shapes by accessing a configuration rules database which is used by the configuration evaluation module to determine whether the configuration parameters of the two functionally interactive shapes in the proposed design violate any configuration rules; and

when a configuration rule is determined to have been violated, automatically evaluating any errors using an error module that accesses a common error database which defines at least some errors as being grouped under a common error, and then returning a common error for display to the user in order to limit the number of error messages presented to the user during the design process;

displaying an icon next to a at least one of the first or second selected shapes to represent at least one common configuration error associated with the shape; and

in response to a user selecting the icon, displaying at least one proposed solution to the common configuration error presented.

REMARKS

Applicants express appreciation to the Examiner for the recent interview held with applicants' representative. As presented herein for reconsideration, the claims have been amended as proposed at the interview. Specifically, claims 1, 3, 4, 8, 11 – 15, 17 – 21 and 23 have been amended, claims 2, 5 – 7 and 24 - 30 have been cancelled without prejudice, and new claim 31 has been presented. Thus, by this paper, claims 1, 3, 4, 8 – 23 and 31 are pending and presented for reconsideration, of which claims 1 (directed to a method) and 31 (directed to a corresponding computer program product) are the independent claims.

As noted by applicants in their specification, when a user is using a design application such as BizTalk Orchestration Designer™, displaying a limited number of messages when encountering configuration parameter errors so that the errors are grouped under common errors allows the designer to more quickly identify and correct the configuration errors. Thus, the "design experience is improved when a user is presented with an abbreviated list of the most important proposed solutions rather than an exhaustive list of all proposed solutions. When presented with multiple proposed solutions, the user must first determine if there is an association between the proposed solutions or a common cause." P. 8 ¶ 27. Thus, automatically identifying a common error for an entire group of errors arising from inconsistent configuration parameters of different software components in accordance with applicants' claimed method and computer program product for implementing the method, significantly simplifies the design process.

As presented herein for reconsideration, the independent claims are directed to a method of alerting a user/software designer to configuration errors that may arise due to inconsistencies between the configuration parameters of one or more of the selected software components for the software design, and a computer program product for implementing the method. The claimed method is implemented in a computing system which includes a visual design surface in the form of a user interface having a plurality of shapes that are selectable by a user, each shape being associated with one or more configuration parameters that define characteristics of software components such as relative positioning of shapes, connections between shapes and other design parameters set by a user, and wherein a user may select and arrange a plurality of the shapes when designing a software application by dragging and dropping them to a design section of the interface. As claimed, the method is comprised of the user selecting a first shape and copying it to the design section of the interface, and then selecting a second shape and copying it to the design section of the interface so that the second shape is functionally interactive with the first shape. Next, the computing system automatically evaluates with a configuration module the configuration parameters of the first and second shapes by accessing a configuration rules database which is used by the configuration evaluation module to determine whether the configuration parameters of the two functionally interactive shapes in the proposed design violate any configuration rules. When a configuration rule is determined to have been violated, the computing system then automatically evaluates any errors using an error module that accesses a common error database which defines at least some errors as being grouped under a common error, and then returns a common error for display to the designer in order to limit the number of error messages presented to the user during the design process. An icon is displayed by the computing system next to at least one of the first or second selected shapes to represent at least one common configuration error associated with the shape, so that in response the designer selects the icon, displaying at least one proposed solution to the common configuration error presented.

In the Office Action the claims were rejected solely on grounds that they were found to be obvious under 35 U.S.C. § 103(a) over U.S. Pat. No. 6,225,998 (Okita et al.) as further modified by teachings found in U.S. Pat. No. 7,165,194 (Paradkar).[1]

Applicants claimed method and computer program product as presented herein for reconsideration are significantly different from the prior art or record and are neither anticipated nor made obvious by the prior art. Okita et al. is directed to a visual design application that relates to improvements in customer transaction processing systems, particularly the visual design of transaction flows. Col. 1 lines 5 – 10. In existing systems, as noted by Okita et al., "generation of routing tables or routing procedures to control transaction flows requires computer programming skills." Recognizing this problem, Okita et al. describe a method for displaying visual primitives of transaction flow used by a transaction processing system as a means by which development of routing tables or routing procedures for customer transaction processing is simplified and better coordinated across distributed environments. Col. 1 lines 49 –

[1] Since Paradkar qualifies as "prior" art, if at all, under 35 U.S.C. 102(e) applicants reserve the right to challenge the status of that reference as qualifying "prior" art. Accordingly, any statement or comment herein to Paradkar is made merely for purposes of argument, and assumes *arguendo* that the reference is proper qualifying prior art.

66 and col. 2 lines 1 – 7. When using the method of Okita et al. to prepare a visual representation of a customer transaction process, "If a particular workflow is not complete, or contains errors, a visual indicator (or visual alert) is generated and displayed by the workflow editor." Col. 15 lines 48 – 50.

As noted in applicants' response of Dec. 11, 2006, and as acknowledged in the present Office Action,[2] "the cited sections of Okita . . . make no mention of 'displaying at least one proposed solution to a configuration error,' as claimed."

Paradkar, on the other hand, is directed to a computer software program used to capture configuration values of predetermined parameters of vendor software and an operating system on which the vendor software is installed. . . . A collateral report of configuration values for the customer computer systems is sent to the vendor for diagnosis of the technical problem experienced by the customer." Abstract. "The support program can be programmed to report configuration errors in the customer environment, and suggest possible solutions." Col. 9 lines 22 – 24.

Apart from whether Paradkar is properly combinable with Okita et al. as asserted,[3] and perhaps more importantly, neither Okita et al. nor Paradkar are directed to simplifying and improving the design experience of a software designer by

> "automatically evaluating with a configuration module the configuration parameters of the first and second shapes by accessing a configuration rules database which is used by the configuration evaluation module to determine whether the configuration parameters of the two functionally interactive shapes in the proposed design violate any configuration rules;
>
> when a configuration rule is determined to have been violated, automatically evaluating any errors using an error module that accesses a common error database which defines at least some errors as being grouped under a common error, and then returning a common error for display to the user in order to limit the number of error messages presented to the user during the design process;

[2] Office Action p. 3 ("Okita fails to explicitly disclose a method of, displaying at least one proposed solution to a configuration error.").

[3] Applicants reserve the right to further challenge whether the Examiner's proposed combination of prior art is properly combinable if and as such becomes a necessary issue to resolution of the rejections, which is not the case since in any event the references fail to teach or suggest the claimed method, as noted above.

displaying an icon next to a at least one of the first or second selected shapes to represent at least one common configuration error associated with the shape; and

in response to a user selecting the icon, displaying at least one proposed solution to the common configuration error presented." Claims 1 and 31.

Thus, for at least the reasons noted, the claims as presented herein for reconsideration are patentable over the prior art of record, and thus favorable action is courteously requested.

In the event the Examiner finds any remaining impediment to allowance of this application that may be clarified through a telephone interview, the Examiner is requested to contact the undersigned attorney.

Dated this ____ day of ________________, 2008.

Respectfully submitted,

RICK D. NYDEGGER
Registration No. 28,651
Attorney for Applicant
Customer No. 047973

RDN:aam
1858765_1